DiffAdv: Generating an Adversarial Example for Any Given Image Using Diffusion Models

Yize CHENG*, Wenbin HU*, Zhuo ZENG*, Yufan DENG*
Hong Kong University of Science of Technology
{ychengbt, whuak, zzengan, ydengbd}@connect.ust.hk

Abstract

*Generating adversarial examples in a hard-label black-box manner is often formulated as an optimization problem. But a significant drawback of these optimization-based attacking algorithms is that a huge number of model queries is required. Therefore, we explore a much more efficient way of crafting adversarial examples using generative models. Particularly, we aim to craft an adversarial example for any given image by leveraging the forward diffusion process and the reverse denoising process of diffusion models. Once this diffusion model is trained, we can use it to generate adversarial examples with no need for querying the model. We name this idea **DiffAdv**. In this project, we validate the soundness of this idea by first training diffusion models using white-box generated adversarial examples and check whether the distribution of adversarial examples can be learnt. Once the soundness of this idea is validated, we can compute the adversarial examples used for training in a hard-label black-box manner and achieve efficient decision-based adversarial attack. Extensive experiments under various attacking scenarios, with different diffusion time steps, and with both single-step and multi-step denoising show that the diffusion model has indeed to some extent learnt the distribution of adversarial examples, resulting in a somewhat effective untargeted adversarial attack. But the attacking performance on targeted attack needs further improvement.*

1. Introduction

Deep Neural Networks (DNNs) have been discovered to be vulnerable to adversarial attacks [2, 8, 19], where the attacker carefully computes an adversarial example that looks almost indistinguishable from the original image, but causes the target model to make a wrong prediction. Crafting adversarial examples, particularly in the most realistic hard-label black-box setting, is a significant way of evaluating and improving the adversarial robustness of modern machine learning models. Various hard-label black-box adversarial attacks [3, 5, 6] have been proposed, but they all suffer from the drawback of requiring a large number of model queries for just a single input, which significantly limits their efficiency and scalability. Therefore, we explore a much more efficient way of computing adversarial examples, that is to train a generative model to learn the distribution of adversarial examples, and sample from that generative model to obtain adversarial examples for any new image. In particular, we leverage the forward diffusion process and reverse denoising process of diffusion models.

The basic idea is to train a diffusion model using a large set of adversarial examples generated by some existing attack, and expect the diffusion model to learn the distribution of those adversarial examples. Once the diffusion model is trained, for any given clean image, we first add noise to it via the forward diffusion process, reaching a stage where the image's original visual semantics are still preserved, but many clean features in the image have been corrupted by the noise added. Subsequently, the reverse denoising process is applied to obtain a "clean image," which is expected to serve as an adversarial example.

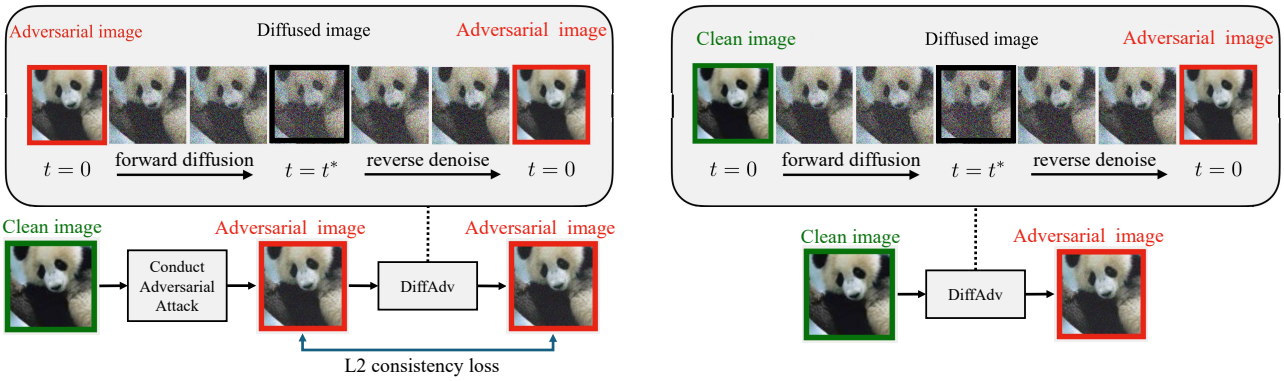
In this project, we conducted extensive experiments to verify the soundness of the idea of DiffAdv. Specifically, we show that in an untargeted attack setting, the diffusion model trained on adversarial examples can produce a perturbed CIFAR-10 test set that results in approximately 15% lower testing accuracy compared to the CIFAR-10 test set generated by a clean diffusion model.

2. Related Works

2.1. Adversarial Attacks

Adversarial attacks exploit vulnerabilities in the model's decision-making process, often by introducing subtle perturbations to input data that are imperceptible to human observers but can cause the model to make incorrect predictions. Since the discovery of adversarial examples by [18], numerous attack methods have been developed. These attacks can be broadly categorized into two settings: white-box and black-box. In the white-box setting, the attacker possesses complete knowledge about the targeted model, including its architecture and parameters. Consequently, the attacker can employ back-propagation to generate adversarial objects, as demonstrated by [8, 13, 15]. In contrast, the black-box setting has gained significant attention

* Equal Contribution.



(a) **Training process** (b) **Inference process**

Figure 1. **Pipeline of DiffAdv**

recently due to its realistic scenario. In this setting, the attacker can only query the model without having direct access to its internal information. Depending on whether the model's feedback provides probability outputs, the attacks can be classified as soft-label attacks or hard-label attacks. In the field of adversarial attacks, different strategies have been proposed in both the soft-label and hard-label settings. The soft-label setting involves techniques such as the ZOO attack [4], which uses finite differences to estimate gradients and performs gradient descent. Other methods include adopting better prior distributions [10], compressing the search space [20], and using gradient-sign-based methods to improve query efficiency [14]. In the hard-label setting, random search methods like the boundary attack have been used to discover adversarial attack [1]. Refinements have been made by proposing superior sampling priors [3] and formulating hard-label attacks within an optimization framework using zeroth-order methods [6].

2.2. Generating Adversarial Examples with Generative Models.

While previous works have proposed successful black-box attacks, their efficiency is hindered by the substantial volume of queries required. A more efficient approach involves leveraging generative models for generating adversarial examples. Several research works have demonstrated the effectiveness of utilizing Generative Adversarial Networks (GANs) specifically for this purpose. In [21], the generator component of a GAN architecture is employed to generate adversarial perturbations, utilizing the information flow from a victim model. To constrain the magnitude of the generated adversarial perturbation, a soft hinge loss is incorporated to bound the L2-norm of the perturbation. Another work [17] generalizes the definition of adversarial examples. It suggests that any inputs that are capable to deceive models without confusing human pose security threats, which is called unrestricted adversarial examples. They utilize GAN structure to generate unrestricted adversarial examples without any clean images. The generated images can successfully fool the victim model and pass human evaluations.

Recently, diffusion models have achieved remarkable advancements in image generation, surpassing models within the GAN family [7]. Therefore, considering the high quality of generated images with diffusion models, it is entirely warranted to delve into generating adversarial examples using diffusion models. However, to the best of our knowledge, no prior research exists on this topic. Our work, on the other hand, showcases the potential and feasibility of employing a diffusion model for the generation of adversarial examples.

3. Methodology

This section presents the technical details of DiffAdv. We adopt the model structure of Improved Denoising Diffusion Probabilistic Models (iDDPM) [16] as diffusion model for DiffAdv. Section 3.1 explains the training process of DiffAdv shown in Figure 1a, and Section 3.2 illustrates how DiffAdv produces adversarial images which are shown in Figure 1b.

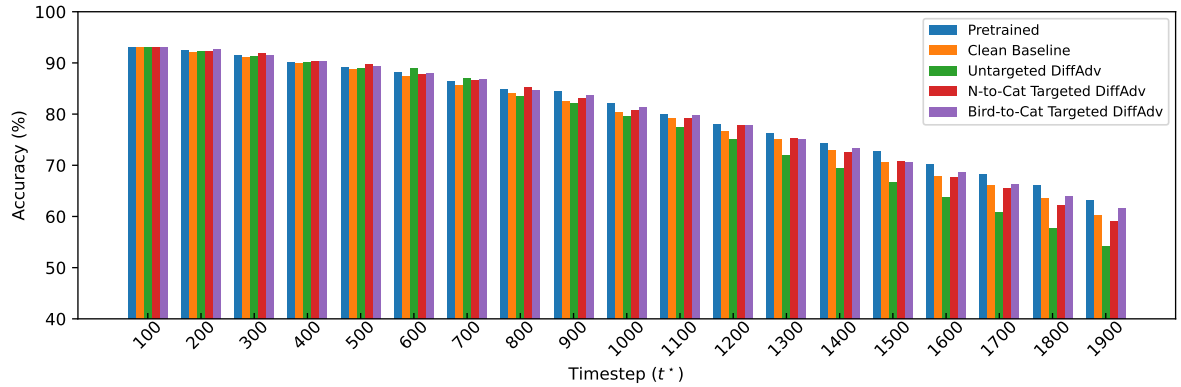
3.1. Training of DiffAdv

DiffAdv is trained with adversarial images, which is obtained from sophisticated adversarial attack method, such as L2PGD attack. These adversarial images are then used to train the DiffAdv following standard diffusion model training. According to [9], with $\alpha_t := 1 - \beta_t, \beta \in (0, 1)$ and $\bar{\alpha}_t := \prod_{s=0}^{t} \alpha_s$, we can obtain x_t from $x_0 \sim D_{adv}$ by

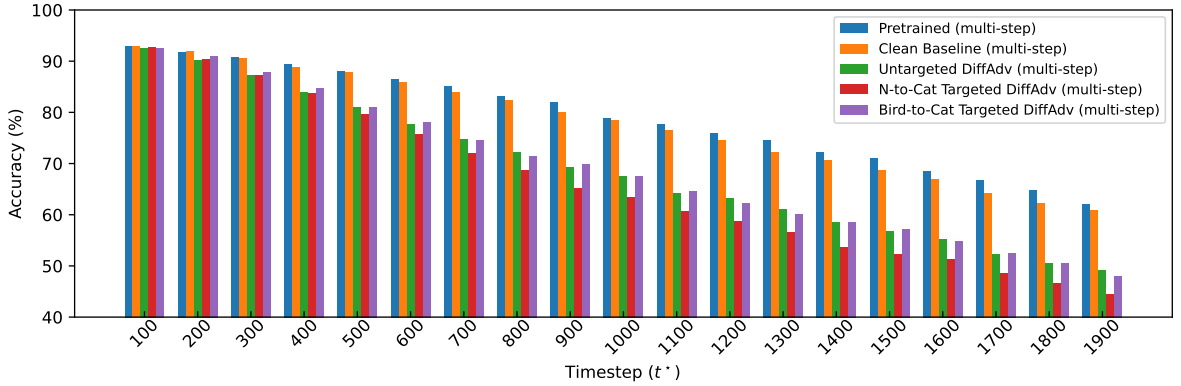
$$\begin{aligned} q(x_t|x_0) &= \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t)\mathbf{I}) \\ x_t &= \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon \end{aligned} \tag{1}$$

where D_{adv} is a set of adversarial examples, and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ is the random noise added to the original image x_0. For each iteration of the training, a time-step $t \in (1, T)$ is randomly chosen, and the input image (x_0) is forward diffused by Equation 1 into x_t.

The actual task that the diffusion model trained on is the denoising step. U-Net, one of the state-of-the-art image-to-image structure, is used to predict the noise ϵ from x_t, and a simple L2 consistency loss is applied between the actual



(a) **Single-step denoising**



(b) **Multi-step denosing**

Figure 2. The **Testing Accuracy (ACC)** following an adversarial attack is evaluated for five different diffusion models, each trained using different methods: [Pretrained] without any additional images, [Clean] using clean CIFAR-10 training set images, [Untargeted DiffAdv] using adversarial examples generated without a specific target class or source class, [N-to-Cat] using adversarial examples generated with the target class set as cat and no source class, and [Bird-to-Cat] using adversarial examples generated with the target class set as cat and the source class set as bird.

noise and predicted noise to train the model:[1]

$$\mathcal{L} = \mathbb{E}_{t,x_0,\epsilon}[\|\epsilon - \epsilon_\theta(x_t, t)\|^2] \tag{2}$$

3.2. Inference of DiffAdv

After finishing the training of DiffAdv, adversarial images can be obtained by letting clean images go through the diffuse and denoise process, which is depicted in Figure 1b. Given a fixed time-step $t^* \in (1, T)$, usually not close to T, the clean images x_0 is diffused by Equation 1 into x_t, and x_t is denoised by the trained U-Net to get the adversarial images $\hat{x}_0$ which is the output of DiffAdv inference.

4. Experiments

4.1. Setup

Dataset. We use the entire CIFAR-10 [12] training set for generating the adversarial examples used for training the diffusion model. During the attack phase, we try generating adversarial examples for the entire CIFAR-10 test set from the trained diffusion model.

Target Classifier. The target classifier, which is the victim model in our experiment, is a ResNet-50 pretrained on CIFAR-10. It can originally achieve 93.65% testing accuracy on the CIFAR-10 test set.

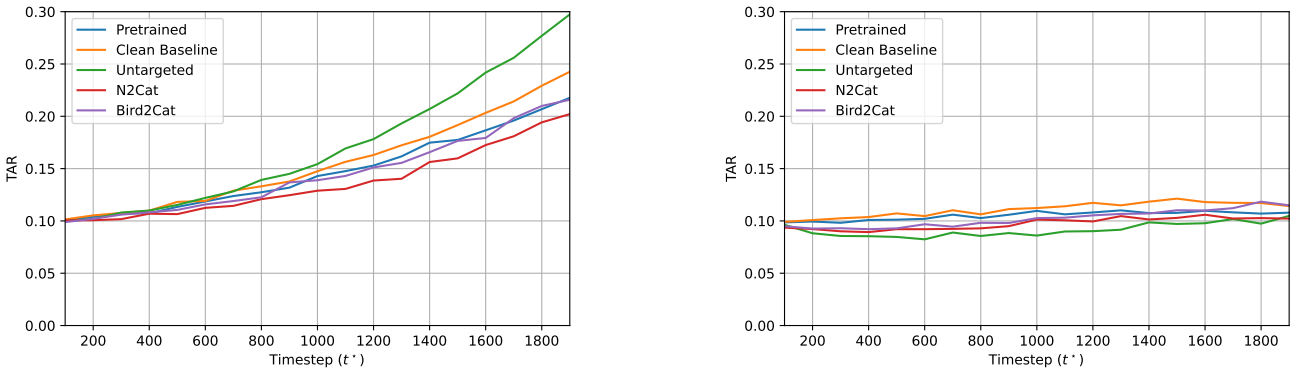
Adversarial Examples for Training. In a vanilla setting for validating the soundness of the idea of DiffAdv, we adopt a simple L2PGD [15] white box adversarial attack for generating the adversarial examples used for training the diffusion model. All adversarial perturbations are bounded by a L2 norm of 5. "Cat" is chosen as the target class in a targeted attack.

Attacking Scenarios. We test DiffAdv under both targeted and untargeted attacks. For targeted attack, we also compare the attacking performance between "N2Cat" and "Bird2Cat", where "N2Cat" refers to the setting that any input clean image is expected to be perturbed towards the target cat, and "Bird2Cat" refers to the setting that an input clean image of a bird should be perturbed towards a cat.

Diffusion Model and Training. We fine-tune the improved denoising diffusion probabilistic model [16] for 10000 iterations using the Adam [11] optimizer from a pretrained checkpoint. The checkpoint was pretrained on CIFAR-10. The learning rate is set to 1e-4. The training settings are identical for all attacking scenarios.

Metrics. To test the effectiveness of the attack, we measure the classification accuracy that the target classifier achieves

[1] For simplicity, we denote $E_{x_0} \equiv E_{x_0 \sim D_{adv}}$



(a) **Single-step denoising** (b) **Multi-step denosing**

Figure 3. **Targeted-class Ratio (TAR)** under five setting mentioned in Fig. 2 with "cat" as target class.

on the perturbed CIFAR-10 test set, where the classification accuracy is defined as:

$$ACC = \frac{\text{\# of correctly classified samples}}{\text{\# of total samples}} \tag{3}$$

In a targeted attack, we define the Target Achieve Rate (TAR) to measure the frequency at which the generated adversarial example is classified as the target class. TAR is defined as:

$$TAR = \frac{\text{\# of adv samples classified as target class}}{\text{\# of total samples}} \tag{4}$$

We expect the classification accuracy to be low on the perturbed CIFAR-10 test sets, and the target achieve rate to be high in a targeted attack.

Denoising Settings. At inference time, we try both single-step denoising and multi-step denoising. When denoising in one single step, after adding noise to the input clean image for $t^\star$ steps, the noisy image $x_{t^\star}$ is directly denoised to a clean image x_0, which is expected to serve as an adversarial examples. When denoising with multiple steps, the $x_{t^\star}$ is iteratively denoised to $x_{t^\star-\delta}, x_{t^\star-2\delta}, \cdots, x_\delta, x_0$.

4.2. Results

The main results are showcased in Figures 2a and 2b, which depict the performance of our target classifier on the perturbed CIFAR-10 test sets generated through single-step denoising and multi-step denoising respectively. The figures reveal that the testing accuracies on the test sets obtained from the pretrained diffusion checkpoint and the diffusion model finetuned on clean CIFAR-10 images are similar, indicating well-defined training settings and a converged denoising model. However, when employing multi-step denoising, the CIFAR-10 test sets generated from diffusion models trained on adversarial examples lead to significantly lower testing accuracies compared to the test sets generated from the pretrained diffusion checkpoint and the diffusion model finetuned on clean CIFAR-10 images. Conversely, the difference in testing accuracy is considerably smaller and negligible when using single-step denoising. Hence, it can be concluded that the diffusion model has indeed to some extent learned the distribution of adversarial examples and can generate adversarial examples for a given clean input image using multi-step denoising.

While the testing accuracies demonstrate the success of the adversarial attack, further analysis reveals that the attack is more effective in untargeted attacks rather than targeted attacks. Under multi-step denoising, which corresponds to successive attacks, the target achieve rate remains stable around 10%, which is the proportion of cat images in the CIFAR-10 test set, regardless of the values of $t^\star$. This indicates that while the attack is successful in an untargeted sense, it does not meet the criteria for a successful targeted attack. Interestingly, we also observe that when using single-step denoising, all diffusion models tend to denoise input images towards a cat image, and this tendency becomes stronger as $t^\star$ increases. Although the exact reason for this behavior is not yet clear, we speculate that it may be due to the fact that the pretrained diffusion checkpoint has been trained to generate cat-like images effectively, and our finetuning process did not eliminate this bias. For a comprehensive illustration of the target achieve rate under both single-step denoising and multi-step denoising, please refer to Figures 3a and 3b respectively.

5. Conclusion

In this project, we propose DiffAdv and explore whether it is possible to leverage the forward diffusion process and reverse denoising process of a diffusion model to generate adversarial examples efficiently. Through extensive experiments, we have demonstrated that the diffusion model is capable of learning the distribution of adversarial examples to some extent. This enables us to leverage the trained model to craft untargeted adversarial examples for any input clean image by employing multi-step denoising. As part of our future work, we plan to reproduce our results by training the diffusion model with adversarial examples generated in a black-box manner. We will also delve deeper into understanding the underlying reasons behind the observed cat bias in the diffusion models and explore ways to enhance the effectiveness of targeted attacks.

References

[1] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models, 2018. 2

[2] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017. 1

[3] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 ieee symposium on security and privacy (sp)*, pages 1277–1294. IEEE, 2020. 1, 2

[4] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, CCS '17. ACM, Nov. 2017. 2

[5] Minhao Cheng, Thong Le, Pin-Yu Chen, Jinfeng Yi, Huan Zhang, and Cho-Jui Hsieh. Query-efficient hard-label black-box attack: An optimization-based approach. *arXiv preprint arXiv:1807.04457*, 2018. 1

[6] Minhao Cheng, Simranjit Singh, Patrick Chen, Pin-Yu Chen, Sijia Liu, and Cho-Jui Hsieh. Sign-opt: A query-efficient hard-label adversarial attack. *arXiv preprint arXiv:1909.10773*, 2019. 1, 2

[7] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021. 2

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[10] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information, 2018. 2

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 3

[13] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world, 2017. 1

[14] Sijia Liu, Pin-Yu Chen, Xiangyi Chen, and Mingyi Hong. signsgd via zeroth-order oracle. In *International Conference on Learning Representations*, 2019. 2

[15] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1, 3

[16] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 2, 3

[17] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models, 2018. 2

[18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. 1

[19] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[20] Chun-Chen Tu, Paishun Ting, Pin-Yu Chen, Sijia Liu, Huan Zhang, Jinfeng Yi, Cho-Jui Hsieh, and Shin-Ming Cheng. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks, 2020. 2

[21] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks, 2019. 2

DiffAdv: Generating an Adversarial Example for Any Given Image Using Diffusion Models

Appendix

A. Visualization of the Generated Adversarial Examples from DiffAdv

Figure 4 presents visualizations of several adversarial examples generated using DiffAdv. A comparison between the denoised outputs from the pretrained diffusion checkpoint and the diffusion model trained on adversarial examples reveals the presence of "adversarial-like" patterns in the latter set of denoised images.

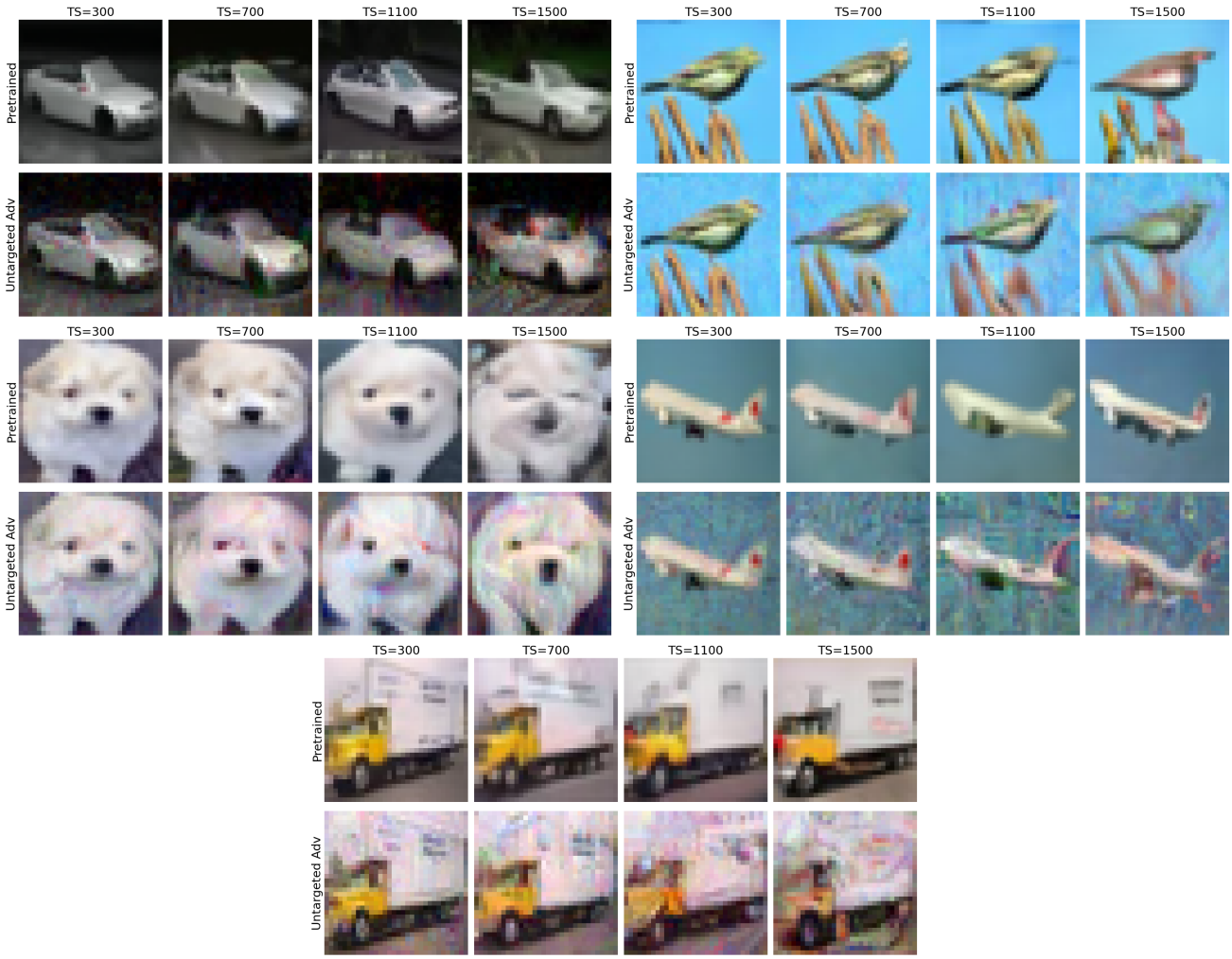


Figure 4. Visualization of the Generated Adversarial Examples